Aalto University
School of Science
Master's Programme in Machine Learning & Data Mining

Henning Lange

Disaggregation by State Inference
A Probabilistic Framework For Non-Intrusive Load Monitoring

Master's Thesis

Berlin, August 30, 2015

Supervisor: Professor Juha Karhunen, Aalto University

Instructor: Professor Mario Bergés, Carnegie Mellon University

Aalto University
School of Science
Degree Programme in Computer Science and Engineering
Master's Programme in Machine Learning

Aalto University
School of Science

ABSTRACT OF THE MASTER'S THESIS

Author: Henning Lange

Title:Disaggregation by State Inference – A probabilistic framework for Non-Intrusive Load Monitoring

Number of pages: 58 | Date: 30.08.2015 | Language:English

Professorship: TIK | Code: T-61

Supervisor: Dr. Juha Karhunen

Advisor:Dr. Mario Bergés

Abstract: Non-intrusive load monitoring (NILM), the problem of disaggregating whole home power measurements into single-appliance measurements, has received increasing attention from the academic community because of its energy saving potentials, however the majority of NILM approaches are either variants of event-based or event-less disaggregation. Event-based approaches are able to capture much information about the transient behavior of appliances but suffer from error-propagation problems whereas event-less approaches are less prone to error-propagation problems but can only incorporate transient information to a small degree. On top of that inference techniques for event-less approaches are either computationally expensive, do not allow to trade off computational time for approximation accuracy or are prone to local minima. This work will contribute three-fold: first an automated way to infer ground truth from single appliance readings is introduced, second an augmentation for event-less approaches is introduced that allows to capture side-channel as well as transient information of change-points, third an inference technique is presented that allows to control the trade-off between computational expense and accuracy. Ultimately, this work will try to put the NILM problem into a probabilistic framework that allows for closing feedback loops between the different stages of event-based NILM approaches, effectively bridging event-less and event-based approaches. The performance of the inference technique is evaluated on a synthetic data set and compared to state-of-the-art approaches. Then the hypothesis that incorporating transient information increases the disaggregation performance is tested on a real-life data set.

Keywords: Energy disaggregation, Non-Intrusive Load Monitoring, Factorial Hidden Markov Model

Preface

The end of this thesis marks my the end of my Masters degree which also marks the end of an Odyssey.
I must admit that academic success was not my number one priority during my Masters but I have learned what I wanted to learn. In the last three years I have changed and grown more than in my previous life combined and ultimately, my drive for knowledge and science is restored after all and I am ready for new academic endeavors.
During my Masters, I have lived in four different countries: Finland, The Netherlands, Germany and the United States. I want to thank everyone who did their best to make me feel at home in the respective countries. Good friends were made, good times were had. Thank you, Finland people. Thanks, Amsterdam. Thanks to the friends I've made at smartB and during the long nights in Berlin. Thanks, CMU guys and especially Mario for his academic and financial support.
I also want to thank my family for their patience and the financial support during my Bachelors and my Masters.

So long, peace out.
Henning Lange

Contents

Chapter 1

Introduction

At the moment fossil fuels are being burned to create more than 83% of the total energy world-wide and the total energy demand has increased by 46% within the last 20 years. As fossil fuel reserves are becoming more and more limited, not only other energy sources need to be tapped into but also means to slow down the increase in energy demand need to be found.[1]

The detrimental impact that producing more energy has on our environment and climate call for effective means to save energy. Studies have shown that if residential consumers are provided with a real time feedback of how much energy their home consumes, 10-15% of energy can be saved. Higher quality feedback namely on the device level could lead to even bigger saving potentials [6]. Since buildings account for roughly 73% of the electrical energy demand in the U.S., the saving potentials of Non-Intrusive Load Monitoring are huge.

Non-intrusive load monitoring tries to algorithmically break up the electricity bill of consumers and provide energy consumption information on a per device level. Not only does this create awareness in consumers (which in itself leads to energy savings) but might also lead to other scenarios in which energy is saved. If e.g. a smart energy system or a consumer is aware that a certain device consumes a certain amount of energy, its runtime could be scheduled to a point in time when exactly this amount of excess energy is available in the grid.

Chapter 2

Approaches for Non-Intrusive Load Monitoring

The idea of Non-Intrusive Load Monitoring was first conceived in [10]. Roughly speaking, it deals with the challenge of inferring the power traces of individual appliances from the overall consumption of a building such that the sum of the individual appliances is approximately equal to the overall consumption. The input to a NILM system is a power trace of a home at the whole-home or at sub-distribution level. Power is inherently an additive quantity that means that if the power is measured at an aggregate level, the superposition of the power draws of the individual appliances is measured. See Figure 1 for a very brief example.

Most of the current NILM approaches can either be classified as event-based or event-less approaches. One major difference between the two is that event-less approaches emphasize explaining the aggregate power trace whereas event-based approaches emphasize explaining events extracted from the power trace. In this chapter the dichotomy between event-less and event-based approaches and the shortcomings of current event-based as well as event-less approaches will be discussed. For a more thorough review of NILM research, see [17] or [18]

2.1 Event-less NILM

Event-less approaches can further be subdivided into two major categories: sparse coding approaches [12] and factorial HMM approaches [13, 11]. Both

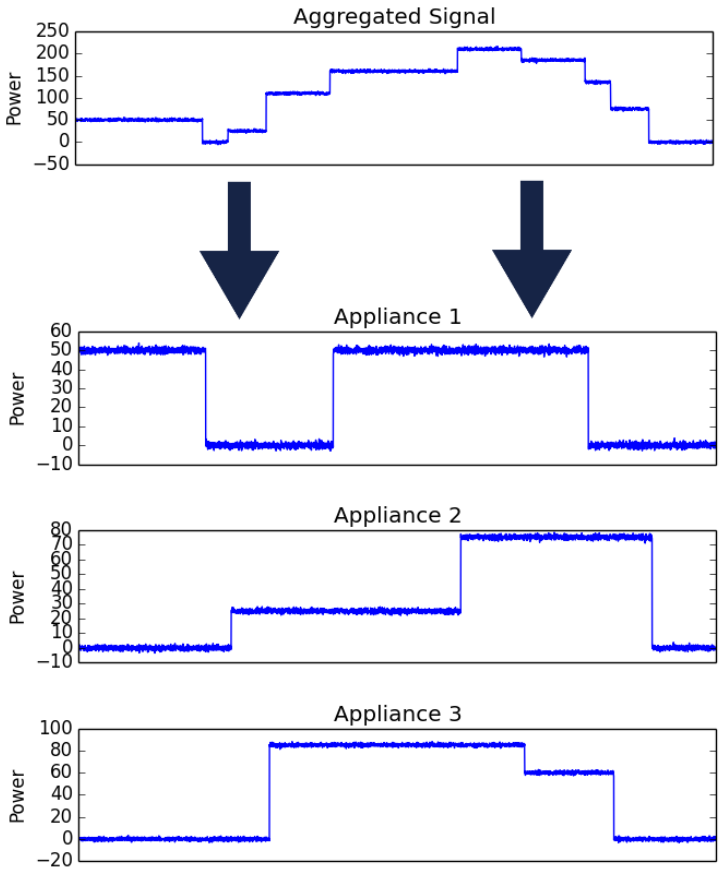

Figure 2.1: Exemplary disaggregation of three devices.

approaches neither rely on event detectors that identify possible state changes in devices nor on classifiers that try to infer appliances given features measured in the power line at time points of possible appliance state changes. The following sections will briefly discuss sparse-coding and FHMM models.

2.1.1 Sparse-coding approaches

This section is based on [12] and will briefly summarizes the main idea behind the approach. It is assumed that the power consumption of individual appliances in a particular house are known. These readings could e.g. be obtained by plug-level sensors. Assume appliance classes such as *refrigerator*,

television or *stove*. One should keep in mind that appliance types encompass appliance instances. Most NILM approaches try to infer the power trace of appliance instances (e.g. the microwave "Breville BMO734XL") whereas this approach tries to infer the power consumption of appliance classes (e.g. microwave). Every appliance class is associated with an index $i \in \{1, 2, ..., k\}$. For every appliance class, assume a matrix $X_i \in R^{T \times m}$. T denotes the length of the power readings, e.g. if a week of data is sampled every minute, then $T = 60 * 24 * 7 = 10080$. m denotes the number of houses for which data is available. Thus, the matrix $\hat{X} = \sum_i^k X_i$ contains the aggregate signal of all houses and the jth row denotes the aggregate signal of house j denoted by $\hat{X}^{(j)}$. The goal is to infer $X_i^{(j)}$ from $\hat{X}^{(j)}$. During training access to each X_i is assumed, thus the approach is a supervised approach.
During training a model for each X_i is trained individually. It is assumed that each X_i can be broken up into a matrix containing the information when each appliance is turned on ($A_i^{n \times m}$) and a matrix containing basis functions ($B_i^{T \times n}$). n specifies the number of basis functions. X_i is then approximated by $X_i \approx B_i A_i$. Additionally, it is assumed that the activation matrix A_i is sparse which can be enforced by adding a regularization term over A_i. Since power is an inherently positive and additive quantity, it can furthermore be assumed that A_i and B_i are positive. During training the joint optimization problem can be stated as:

$$A_i^*, B_i^* = \underset{A_i \geq 0, B_i \geq 0}{\operatorname{argmin}} \frac{1}{2} ||X_i - B_i A_i||_F^2 + \lambda \sum_{p,q} (A_i)_{p,q} \qquad \text{subject to}$$

$$||b_i^{(j)}||_2 \leq 1$$

$||M||_F = \sqrt{(\sum_{p,q} M_{p,q})}$ denotes the Frobenius norm, whereas $||V||_2 = \sqrt{\sum_p V_p^2}$ denotes the ℓ_2 norm. The joint optimization problem is however not convex but optimizing over either B_i is convex if A_i is held fixed and vice versa. Thus, the algorithm to obtain A_i and B_i alternates between optimizing A_i and B_i.

During inference, given the aggregated signal $\bar{X} \in \mathbb{R}_+$, the optimal activations are sought that minimize:

$$A_i' = \underset{A_i \geq 0}{\operatorname{argmin}} \frac{1}{2}||\bar{X} - \sum_i B_i^* A_i||_F^2 + \lambda \sum_{p,q} (A_i)_{p,q}$$

For the estimate of the power consumption of appliance i in building j then holds: $(X_i')^{(j)} = (B_i^* A_i')^{(j)}$. This convex optimization problem can be solved by concatenating all activation and basis function matrices into two matrices A and B^*.

One should note that the matrix containing basis functions B_i is obtained under the constraint that $||b_i^{(j)}||_2 \leq 1$. This basically means that the matrix containing the activations A_i essentially contain the information on how much an appliance actually consumes. The matrix B_i contains so to say temporal usage patterns of appliances, i.e. a stove is mostly turned on for breakfast, lunch and dinner. During inference these patterns are exploited. The actual information about how much appliances consume is neglected during inference. This means that the model might confuse appliance with very different power levels but similar temporal activation patterns.

2.1.2 Factorial Hidden Markov Models

Recent advances in NILM have employed Additive Factorial Hidden Markov Models to disaggregate energy[13]. In this section Hidden Markov Models and their distributed generalization Factorial Hidden Markov Models [9] will be introduced.

Hidden Markov Models

Hidden Markov models have emerged to be one of the most successful tools for modeling discrete and continuous time series. An HMM is a mixture model that encodes historical information in a single multinomial variable which we call the hidden state. The hidden state can take K many discrete values. Let s_t denote the hidden state at time t. At every point in time an observation y_t is emitted depending on the state of the hidden variable. Two independence assumptions are made that make training and inference tractable:

1. The observation y_t is independent of all other observations given the state of the hidden variable s_t

2. The hidden variable s_t is independent from $s_1, ..., s_{t-2}$ given the state of the hidden state s_{t-1} (first order Markov assumption)

Because of these independence assumptions the joint probability of an observation sequence $Y = y_1, ..., y_T$ and the hidden state sequence $S = s_1, ..., s_T$ can be rewritten:

$$\begin{aligned} P(Y, S) &= P(Y|S)P(S) \\ &= P(y_1, ..., y_T|s_1, ..., s_T)P(s_1, ..., s_T) \\ &= P(s_1)P(y_1|s_1)\prod_{t=2}^{T} P(s_t|s_{t-1})P(y_t|s_t) \end{aligned}$$

The state transition probabilities $P(s_t|s_{t-1})$ can be encoded in a $K \times K$ state transition matrix. Depending on the nature of the observation, different models of $P(y_t|s_t)$ can be employed. If for example there are D-many discrete observations, $P(y_t|s_t)$ can be encoded by a $K \times D$ emission matrix. For continuous emissions (observations), $P(y_t|s_t)$ could in principle be modeled by any distribution or even neural networks. [9]
Due to its multinomial nature, efficient algorithms for supervised inference, namely the Viterbi [15], and for unsupervised parameter estimation, namely Baum-Welch [3], are available.

However, Hidden Markov Models become intractable if a single hidden state has to incorporate multiple phenomena. Consider the following scenario: there are two dice. In every point in time, both dice are flipped to one side. Because they are not rerolled but simply flipped to one side, it is for example impossible for one dice that 6 is facing up after 1 faced up. What we observe is not the individual dice but a function of the sides that are facing up. If we were to model this simple example with a traditional Hidden Markov Model, there are $6^2 = 36$ possible hidden states with a state transition probability matrix with $36^2 = 1296$ entries. If the number of dice involved increases, the state space of the hidden variable increases exponentially. The size of the state transition probability matrix grows even faster. Factorial Hidden Markov Models are a generalization that try to tackle this problem.

Factorial Hidden Markov Models

Factorial Hidden Markov Models (FHMM) are a generalization of Hidden Markov Models in the sense that numerous hidden states evolve in parallel. In this work we only consider the case where each hidden state is decoupled from the other hidden states, i.e. they all evolve independently. For the dice example above this would mean that there is one hidden state for each dice. Thus, $s_t = s_t^1, ..., s_t^N$ for N many dice or HMM chains. Due to the distribution of the hidden state, the parameter space of FHMMs is much smaller compared to modeling the same phenomenon with a traditional HMM: for the FHMM there are N many state transition probability matrices of size 36, whereas for traditional HMM a state transition probability matrix of the size 6^{2N} would have to be estimated.[1]

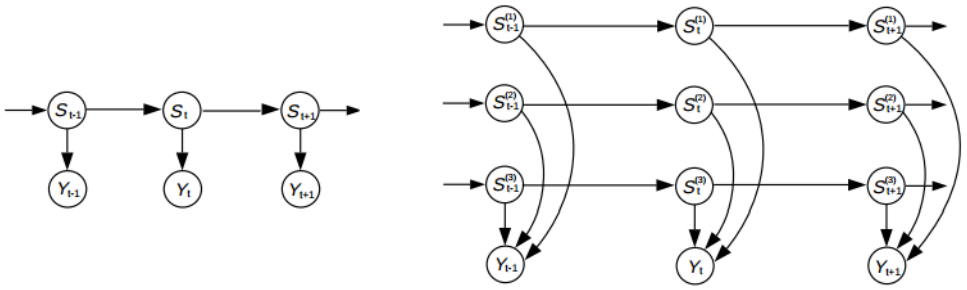

Figure 2.2: Taken from [9]. A depiction of the conditional independence of a traditional HMM (left) and a factorial HMM (right) in a graphical model.

If marginal independence of the individual hidden states is assumed, then the joint probability can be rewritten as:

[1] To get a feel for how fast the state transition probability matrix grows: even for just 4 dice a traditional HMM would require more than 1.6 million parameters, whereas FHMM would only require to model 144 parameters.

$$
\begin{aligned}
P(Y,S) &= P(Y|S)P(S) \\
&= P(y_1,...,y_T|s_1,...,s_T)P(s_1,...,s_T) \\
&= P(s_1)P(y_1|s_1)\prod_{t=2}^{T} P(s_t|s_{t-1})P(y_t|s_t) \\
&= P(s_1)P(y_1|s_1)\prod_{t=2}^{T}\prod_{n}^{N} P(s_t^n|s_{t-1}^n)P(y_t|s_t)
\end{aligned}
$$

The observation y_t is dependent on the joint state s_t. In the dice-example from above, $P(y_t = \sum_n^N s_t^n|s_t) = 1$. If the observation is however discrete then the joint state could influence the $K \times D$ emission probability matrix, whereas if the observation is Gaussian distributed, every state could be associated with parameter $\mu_{s_t^a}$ and the emission probability could be defined as:

$$
P(y_t|s_t) = N(y_t|\sum_{n}^{N} \mu_{s_t^a}, I)
$$

with I being the identity matrix[2] and $N(y|\mu, I)$ being the multivariate Gaussian distribution.

So far, we have seen that Factorial Hidden Markov Models can reduce the otherwise exponential parameter space by decoupling states. However, the hidden states at a single point in time become conditionally dependent given an observation y_t: Consider again the dice example and assume there are two dice. Let's assume we have observed the emission $y_t = 3$. If the hidden states were conditionally independent then it would hold that $P(x_t^1|y_t = 3, x_t^2) = P(x_t^1|y_t = 3)$ but this is obviously not the case, since it must hold that $x_t^1 + x_t^2 = 3$. The inference problem for FHMM tries to infer the most probable hidden state sequence given an observation sequence. The fact that

[2]For the sake of simplicity. The covariance matrix could of course also be dependent on s_t or be some other constant.

the hidden states encompassed in a single joint state become conditionally dependent given an observation makes exact inference intractable. Exact inference would require looping over exponentially many settings of joint states.

Gibbs sampling for inference

Since computing the exact posterior distribution is computationally intractable, approximate inference techniques are required for inference. The posterior distribution can be approximated by Monte Carlo sampling procedures. In this work we will consider one of the simplest sampling techniques: Gibbs sampling[8]. For this the hidden states sequence S is initialized randomly and in each step of the sampling procedure, each state is updated stochastically conditioned on the observations and all other hidden states. Due to the independence assumptions it is sufficient to sample s_t^n from:

$$\begin{aligned} &P(s_t^n | s_t^1, ..., s_t^{n-1}, s_t^{n+1}, ..., s_t^N, s_{t-1}^n, s_{t+1}^n, y_t) \\ \propto &P(s_t^n | s_{t-1}^n) P(s_{t+1}^n | s_t^n) P(y_t | s_t) \end{aligned}$$

Sampling every hidden state once results in computation in $\mathcal{O}(TNK)$ operations and the sequence of the overall state sequence S defines a Markov chain over the state space model. If all probabilities are bounded away from 0, then it can be shown that this Markov chain approximates the true posterior distribution $P(S|Y)$.

Factorized variational inference

The idea behind completely factorized variational inference is that the posterior distribution $P(S|Y)$ can be approximated by a tractable distribution $Q(S)$. $Q(S)$ provides a lower bound on the log likelihood. Mathematically speaking:

$$
\begin{aligned}
P(y_t) &= \sum_{s_t} P(s_t, y_t) \\
logP(y_t) &= log \sum_{s_t} P(s_t, y_t) \\
&= log \sum_{s_t} Q(s_t) \frac{P(s_t, y_t)}{Q(s_t)} \\
&\geq \sum_{s_t} Q(s_t) log \frac{P(s_t, y_t)}{Q(s_t)} \qquad \text{(Jensen inequality)}
\end{aligned}
$$

The idea is to minimize the difference between the tractable distribution Q and the true distribution P. As a model of Q one can use a model whose independence structure is tractable by for example removing the conditional dependence between the hidden states given the observation. Inference is then made on the simpler and tractable Q-distribution. As a difference measure for the similarity between the Q and P distribution the Kullback-Leibner divergence is often chosen because:

$$
\begin{aligned}
KL(Q||P) &= \sum_{s_t} Q(s_t) log \frac{Q(s_t)}{P(s_t|y_t)} \\
&= \sum_{s_t} Q(s_t) log \frac{Q(s_t)}{P(s_t, y_t)} + log(P(y_t))
\end{aligned}
$$

This equation is exactly the difference between the right-hand and left-hand side of the equation above. Since $P(y_t)$ is independent from Q, minimizing the KL-divergence means minimizing $\sum_{s_t} Q(s_t) log \frac{Q(s_t)}{P(s_t, y_t)}$.

As already stated above, in order for this technique to make inference tractable, a suitable model for $Q(s_t)$ has to be chosen that eliminates some of the dependence structure in the P-distribution. The parameters (so called variational parameters) of Q are chosen in such a way that the difference to the true P distribution is minimized. This means that the variational parameters somehow have to encode the dependence structure of the true posterior

distribution. Thus, the parameters of the P distribution are in a sense partitioned based on the dependence structure chosen for the Q-distribution and minimizing the KL-divergence between the Q and P distribution will introduce circular dependencies between the parameters. Therefore approximating P by Q becomes an iterative algorithm much like Expectation Maximization. However, if both the P- and Q-distribution stem from the *exponential family*, it was shown that the resulting algorithm will converge to a global maximum.

Additive Factorial Hidden Markov Models

Recent advances in NILM have employed Additive Factorial Hidden Markov Models to disaggregate energy[13]. These models are factorial in the sense that the state factors into independent chains and the model emits the output of an additive function of all hidden states. See Figure 2.3 for a graphical representation of the model. The dynamics of each appliance are modeled by a single chain.

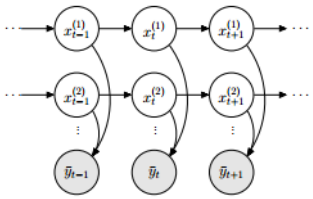

Figure 2.3: Taken from [13]. The additive FHMM model.

Exact inference is computationally intractable in such a model since the space of possible states is exponential. Every chain in the FHMM model evolves marginally independent, thus if there are A many chains (appliances) and each appliance can be in k states then exact inference would require to loop over k^A states. The exponential nature of this problem is tackled by making the *one-at-a-time* assumption which states that maximal one appliance can change states at any point in time. If the sampling frequency is high enough, this assumption is rarely violated.
The emissions are modeled by a Gaussian distribution whose mean is dependent on the states of the individual chains: let μ_j^a denote the mean emission of appliance a in state j, x_t^a be the state of appliance a at time t, $x_t = x_t^1, ..., x_t^A$

and y_t the aggregate emission, then

$$P(y_t|x_t) = N(\sum_{a}^{A} \mu^a_{x^a_t}, \Sigma)$$

The hidden states x evolve like traditional HMMs.
Assume an appliance changed state at time t, the quantity of power change $\Delta y_t = y_t - y_{t-1}$ might hold much information about the appliance that actually changed state, thus it is sensible to also model the difference

$$P(\Delta y_t|x_t, x_{t-1}) = N(\sum_{a}^{A} \mu^a_{x^a_t} - \mu^a_{x^a_{t-1}}, \Sigma)$$

The difference model is good at capturing the changes of the y_t that might help identify the identity of appliance state changes as it explicitly models the increase in power but in isolation errors might accumulate over time.
Gaussian distributions are naturally prone to outliers and outliers in the application domain of energy disaggregation are quite common as e.g. an unmodeled appliance is turned on. Also some appliances like for example light switches produce surges in power when turned on that ultimately lead to spikes in the aggregate signal. These spikes might cause a purely Gaussian model to receive a probability of 0. In order to alleviate this problem, a "robust mixture component" z is introduced. This component can take arbitrary values but its probability is constrained. Two consecutive values of the robust mixture component are constrained in that:

$$P(z) = \frac{1}{Z(\lambda, T)} exp[-\lambda \sum_{t}^{T} |z_t - z_{t-1}|]$$

This mixture component relaxes the Gaussian distribution that models the aggregate emission:

$$P(y_t|x_t, z_t) = N(\sum_{a}^{A} \mu^a_{x^a_t} + \Sigma^{-1} z_t, \Sigma)$$

Note that the output y_t might be multi-dimensional, thus z_t is multiplied by Σ^{-1} such that it commensurates with y_t. Analogically, the difference model is also augmented with a robust mixture component:

$$P(\Delta z) = \frac{1}{Z(\lambda, T)} exp[-\lambda \sum_{t}^{T} |z_t|]$$

$$P(\Delta y_t | x_t, x_{t-1}, \Delta z_t) = N(\sum_{a}^{A} \mu^a_{x^a_t} - \mu^a_{x^a_{t-1}} + \Sigma^{-1} \Delta z_t, \Sigma)$$

The λ-parameter controls how much of the signal can be absorbed by the robust mixture components. If λ is too small it might fail to absorb the outliers, if however λ is too big, all energy will be absorbed by z_t.

In [13] the authors note that the posterior distribution of the difference and the additive model must agree but that the values the robust mixture models are not enforced to agree. Thus for the posterior distribution holds:

$$P(X|Y) = \prod_{t}^{T} P(x_t|x_{t-1}) P(y_t|x_t, z_t) P(z) P(\Delta y_t | x_t, x_{t-1}, \Delta z_t) P(\Delta z)$$

Inference by Integer Programming

The idea behind inferring the most probable state sequence with Integer Programming is to reformulate the problem as an integer programming problem. In this section we will briefly cover how to reformulate inference of a traditional HMM as such a problem and refer to the rather long derivation in [13] for how this can done for a FHMM.
As we have seen earlier, the posterior distribution of a HMM can be rewritten as:

$$P(X|Y) = P(x_1) P(y_1|x_1) \prod_{t=2}^{T} P(x_t|x_{t-1}) P(y_t|x_t)$$

thus:

$$log(P(X|Y)) = log(P(x_1)) + log(P(y_1|x_1)) + \sum_{t=2}^{T} log(P(x_t|x_{t-1})) + log(P(y_t|x_t))$$

Now, variables are introduced that can take integer values. We will drop the first two terms for mathematical convenience but the strategy is analogous. Let K be the number of states:

$$\sum_{t=2,j,i}^{T,K,K} Q(x_t, x_{t-1})_{i,j} log(P(x_t|x_{t-1})) + \sum_{i}^{K} Q(x_t)_i log(P(y_t|x_t))$$

In order to define a proper sequence through the state space, these constraint have to be met:

$$\sum_i Q(s_t)_i = 1$$
$$\sum_i Q(s_t, s_{t-1})_{i,j} = 1$$
$$\sum_i Q(s_t, s_{t-1})_{j,i} = 1$$
$$Q \in \{0, 1\}$$

The idea is to now relax the integer constraint and solve the then convex problem. One should however note that this is not possible without the one-at-a-time constraint for FHMMs.

2.1.3 Short comings of event-less approaches

Some appliances exhibit very characteristic behaviors during state changes, so called transients. For example, recent studies have shown that high frequency information during state changes, so called *harmonics* can improve classification performance in event-based approaches [5]. Also more coarse quantities like the long-term shape of the transient on a low resolution can be very characteristic for some appliances and aid disaggregation. Figure 2.4 shows two exemplary long-term low-resolution transients of a garage door (left) and a fridge (right). The garage door exhibits a very characteristic

power trace during its activation: the power will rise to $\approx 500W$ twice before going back to $\approx 100W$, the refrigerator on the other hand exhibits a short $\approx 500W$ spike before consuming $\approx 120W$. Because of the Markov assumption of most purely event-less approaches, proper use of these long-term dependencies cannot be made.

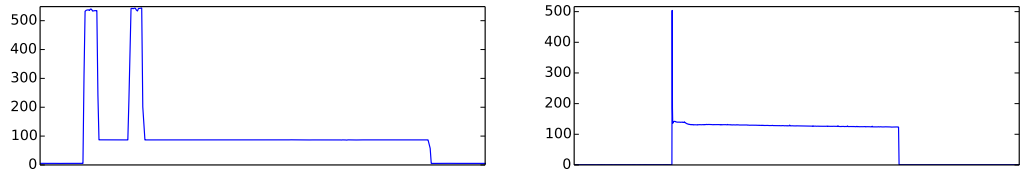

Figure 2.4: Characteristic long-term low-resolution shapes of two appliances. Garage door (left) and refrigerator (right)

On top of that, the NILM problem is inherently an ill-posed problem as it is a one-to-many mapping: a single power trace is used to infer the power trace of multiple appliances. Approaches that can incorporate side-channel information like e.g. light intensity, sound or electro-magnetic radiation might alleviate this problem but making use of this information in classical event-less approaches is not straight forward.

From a computational point of view, most event-less approaches try to explain power at every point in time and neglect the fact that if no significant change in power was detected, a state change of an appliance is unlikely. Event-less approaches based on factorial HMMs rely either on integer programming techniques or sampling methods for inference. If the posterior is not constrained in some way, sampling methods are prone to run into local minima: assume an appliance that consumes $1000W$ was turned on and the state of a 2-state appliance is sampled that consumes $500W$. Without constraining the posterior (e.g. that only a single appliance can change its state at every point in time), sampling techniques might partially explain away the power and when the state of the appliance that is actually responsible for the increase in power is sampled, the residual unexplained power might not be sufficient. Mean-field as well as (Block) Gibbs sampling techniques have shown to be susceptible to local minima and even though quadratic integer programming techniques reformulate inference as a convex approximate inference problem, these approaches lack the ability to trade computational

time for improved approximation accuracy.

2.2 Event-based NILM

The majority of event-based NILM approaches consist of four stages: event detection, feature extraction, classification and energy estimation. In the event detection stage points in time with a statistically significant change of the aggregate power consumption are identified. These changes hopefully or most probably signify a change of the power draw of one of the appliances that are being measured. An event is thus merely a time stamp: a point in time where something happened.
The feature extraction and classification stage try to infer which appliance evoked the events that were detected in the event detection phase. A feature extractor typically scans the proximity of an event and extracts quantities that might shed light on to which appliance's power load changed: e.g. different devices consume different amounts of power. If a light-bulb is turned on the change in power load will change to a smaller degree to when say a water boiler is turned on, thus the difference between the pre-event compared to the post-event power load might give information about the appliance that caused the event. Some appliances exhibit special characteristics when their state is changed, e.g. old light switches might exhibit a sudden power spike (called power surge) when a light bulb is turned on. Recent research has shown that high frequency oscillations (so called *harmonics*) seem to improve classification performance [5]. However, incorporating harmonic features into classification requires high sampling frequencies of the underlying data.
Some features might even stem from an external source: if an unknown appliance is turned on at 8am it is much more likely to be a coffee machine than a television set. External features such as temperature, season or time of the day might play an important role in energy disaggregation. It is the feature extractors job to identify those quantities that help explain the identity of the appliance that caused the event.
In the classification stage the features of the events are being fed into a classifier that ultimately tries to associate an event with an appliance. Many appliance are so-called multi-state appliances. A simple example of a multi-state appliance is a fan whose fan speed can be set into more than two settings. The power load in the different settings might differ and since the ultimate goal is to infer how much power a device is consuming, the informa-

tion of state changes is vital for many energy estimation algorithms. That is why many classifiers not only identify the appliance that caused an event but also the post-event state of this appliance.
In the last stage, a power trace of each individual device is tried to be reconstructed. If for example all state changes of a fan and the power loads of the fan in all its states are known, reconstructing a power trace becomes trivial. In real-world scenarios this is however not always the case and certain constraints have to be met: the sum of the power traces of the individual appliance should to some degree approximate the aggregated signal.

2.2.1 Short-comings of purely feed-forward methods

Most NILM approaches are purely feed-forward, i.e. first events are extracted and the decision about events is then final even if the classifier is then forced to make a very low probability decision on some events. There is no possibility of the classifier to signal back to the event detector that the event detector might have made a mistake. Furthermore, the energy estimation phase cannot feed information back into the event-detection and classification stage: if for example the sum of the power traces of the individual devices greatly exceeds the aggregate power then something has probably gone wrong.

Event-based approaches often extract features of points of interest (events) and then neglect the aggregate signal. In comparison, event-based approaches try to explain features of the aggregate power trace whereas event-less approaches often try to explain the aggregate power itself. Both approaches dismiss information.

Another downside of current event-based approaches is that they are feed-forward. In order to understand why closing the feedback loop between the different stages of disaggregation might greatly improve on the performance of event-based NILM systems, two observations must be made:

1. In a purely feed-forward system, an error in earlier stages will confuse later stages.

2. Neither event-detection nor classification can yield "perfect" results.

Point (1.) can be explained by a very simple example: assume an oversensitive event detector that detects an appliance state change even though

there was none. As long as the feature extraction and classification stage are forced to associate an appliance with said event, the energy estimation stage of the system will have to deal with a nonsensical state transition. In this scenario the classification stage is working fine, just the absence of a feedback-loop to the event detection from either the energy estimation or the classification forces the classification stage to make mistakes.
Point (2.) does not need much explanation: event detectors as well as classifiers are machine learning algorithms whose output is and probably always will be noisy.

This work will try to bridge event-based with event-less approaches. From the perspective of event-based approaches, the stages of disaggregation are tied together into a single probabilistic framework that allows for sub-optimal performance of the individual components. From the perspective of event-less approaches, a framework will be presented that allows for efficient approximate inference in Factorial Hidden Markov Models allowing for the incorporation of high-level and external features.

Chapter 3

Data collection for NILM

In this work the performance of the proposed probabilistic framework will be evaluated on a real world data set. Collecting data to evaluate NILM systems is not a trivial problem and in the following chapter different design choices for the data collection will be described.

3.1 Design choices

In order to assess the performance of NILM algorithms, the systems outputs need to be compared to ground truth and ground truth data needs to be collected. This is not a trivial problem because a number of sensors need to be installed and some feasibility trade-offs need to be made: plug-wise sensors are small and cheap and can be installed to sub-meter individual appliances but often only offer low temporal resolution, i.e. sampling rates in the range between 0.5 to 2Hz. Sensors that collect data at higher frequencies (2kHz+) are often more expensive and big and can effectively only be used to measure data at either a whole-home or sub-distribution level.
There is, so to say, always a trade-off between temporal and spatial resolution. If many appliances are to be sub-metered (high spatial resolution), one has to revert to low-cost low-resolution meters (low temporal resolution).
Higher sampling rates allow the extraction of richer and higher resolution features that might aide the classification stage but also demand more storage and might cause data throughput problems when collecting the data.
Other questions that influence the design of a data collection setup might be what the data needs to offer in order to be of use for evaluation. When eval-

uating NILM algorithms the question arises what to actually evaluate. If e.g. the event detection stage should be assessed then ground truth information about state changes needs to be provided. If however the energy estimation phase is to be assessed, ground truth about either the energy consumption or knowledge about the power traces of individual appliances need to be available.
On top of that, different NILM algorithms might make different assumption about the underlying data: some algorithms may require that all devices that are being measured within the aggregated signal are known. This requires all devices to be sub-metered on a per device level.

3.1.1 Challenges in sub-metering

Most public datasets like for example the REDD dataset [14] or ECO [4] measure the current and voltage at the sub-distribution level with a high-frequency meter (16kHz in the case of REDD) and sub-meter most appliances or groups of appliances with plug-level meters (0.5Hz in the case of REDD). This technique seems to provide the best of two-worlds since high resolution features can be extracted by the high-resolution meters while still providing power traces of the individual appliances. However, numerous problems arise when employing this technique. First, there are time synchronization problems: the clocks of all meters must be synchronized. The higher the sampling frequency, the harder this problem becomes. Second, most NILM algorithms make the assumption that the sum of the sub-metered appliances is equal to the measurements of the sub-distribution meter but this is seldom the case. The sum of the sub-metered appliances in the REDD as well as ECO dataset differ from the aggregate signal. Surprisingly, the difference between the sum and the aggregate changes sign, i.e. it is not the case that some appliances were simply not sub-metered but there might be calibration errors. Third, even to capture a single event for some appliances that are used seldom like for example an air compressor, the data collection system has to be up and running for sometimes weeks. It is very likely that during that time some of the meters fail which ultimately leads to data loss.

Chapter 4

Algorithmic Ground Truth Creation

In order to evaluate and train the different stages of a NILM system, different types of ground truth data are required. Evaluation of event detection requires ground truth about events whereas evaluating classification requires labeled events. Event detection is predominately assessed by computing Precision & Recall [7] whereas the performance of the classification stage is rated by the percentage of correctly classified events. The energy estimation phase can be assessed by different levels of granularity and so far no single performance criterion has become the standard. Some apply the idea of precision and recall to the power traces of the individual appliances. In this context, recall measures what portion of the energy that was assigned to an appliance is correctly classified and precision indicates the portion of the total energy assigned to an appliance, that truly belonged to that appliance. Let y_t be the actual power and $\hat{y}_t$ its prediction, then:

$$[x]_+ = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{else} \end{cases}$$

$$prec(y^a, \hat{y}^a) = 1 - \frac{\sum_t^T [y_t^a - \hat{y}_t^a]_+}{\sum_t^T y_t^a}$$

$$recall(y^a, \hat{y}^a) = 1 - \frac{\sum_t^T [\hat{y}_t^a - y_t^a]_+}{\sum_t^T \hat{y}_t^a}$$

Precision and recall can be combined into the F_1-score:

$$F_1 = 2\frac{prec \cdot recall}{prec + recall}$$

Precision and recall allow to identify if an appliance was assigned too much energy to or too little but it does not allow to assess the overall performance well. One could potentially take the mean of all F_1-scores for all appliances, this would however not take into account that the energy consumed by some appliances is much smaller than the energy consumed by other appliances.
The *disaggregation error* allows to aggregate the performance on different devices more gracefully:

$$de = \sqrt{\frac{\sum_{a,t} \|y_t^a - \hat{y}_t^a\|_2^2}{\sum_{a,t} \|y_t^a\|_2^2}}$$

A very related measurement that we will use in the context of quantization of power time series is the *mean deviation* and the *quantization error*:

$$q\_error = \frac{\sum_t |y_t^a - \hat{y}_t^a|}{\sum_t y_t^a}$$

$$MD = \frac{1}{T}\sum_t^T |y_t^a - \hat{y}_t^a|$$

The datasets considered in this work contain power traces of the individual appliances and whole-home power readings. This means that ground truth to assess the energy estimation stage is provided. However additional information needs to be extracted before the event detection and classification stages can be trained and evaluated. One way to extract this additional information would be to manually annotate the single appliance recordings with events. This is however not feasible if either the number of appliances or the number of events is big. A principled and automated way to extract ground truth from single appliance power traces is sought.

4.1 Events

The datasets contain power traces of the individual appliances. However it is lacking events and state transition information in order to be of use for the probabilistic framework proposed in chapter 5. Simplifying assumptions are made that can in a sense be understood as quantization. The single appliance measurement are approximated by a piece-wise constant function. It is important to understand that this introduces an error. This error is referred to as the quantization error. The following assumptions are made:

1. Every appliance is a multi-state device with k_i states with $k_i > 1$. The number of states differs from device to device.

2. The mean power consumption in every state is constant (first order stationarity). Assuming first-order stationarity introduces a quantization error. The quantization error should not exceed ϵ.

3. An event signifies the state change of an appliance.

These assumptions allow us to automatically create ground truth on events and states for a reading of a single appliance in an unsupervised fashion: in a first step an event detector is applied to the aggregated signal. Let this set of global events be denoted by $E^g = \{e_0^g, ..., e_N^g\}$. Thus, all appliances share the same events, however the state transition information (label of the event) that will be inferred in a subsequent step differs for each appliance. Also, to reduce the computational cost, a less sensitive local event detector is applied to the power trace of each appliance: let the set of local events for each appliance a be denoted by $E^a = \{e_0^a, ..., e_{N_a}^a\}$. Every event that is not present in the set of events detected by the global event detector (applied to the aggregate signal) is removed from the local events. Also, for the sake of algorithmic convenience assume that the time stamp of the first and last power measurement is added to each E^a. Let the resulting set of locally detected events on the power trace of appliance a be denoted by $E^a \leftarrow (E^a \cap E^g) \cup \{0, T\}$. This step has little influence on the state transition information ultimately inferred but it reduces the computational costs substantially. Keep in mind that the set of local events differ from appliance to appliance.

4.2 Event labels and power levels

A recursive and greedy algorithm is employed in order to infer the number and the power levels of states as well as the state transition information of the events. Initially, it is assumed that every event marks a state transition into a new unknown state. Thus, it is assumed that every appliance has as many states as there are consecutive pairs of events. In every iteration of the algorithm, two states are merged into a single state sharing the same power level. Unless the power level of two states is exactly equal, every merging increases the quantization error. The simplest description of the appliance is sought that is why as many states as possible are merged whilst not exceeding a threshold in quantization error. The algorithm described here is a type of *hierarchical agglomerative clustering*[16].

For this, a tuple g is created for every consecutive pair of events. Let p_t^a be the power measurement of appliance a at time point t. The tuple contains the duration and the mean power consumption of the intra-event state. Every tuple constitutes a state and the mean power consumption is an estimation of the power consumption during that state.

$$g_n^a = (\rho_n^a, \tau_n^a) \tag{1}$$

$$\tau_n^a = (e_{n+1}^a - e_n^a) \qquad \text{(state duration)}$$

$$\rho_n^a = (e_{n+1}^a - e_n^a)^{-1} \sum_{t=e_n^a}^{e_{n+1}^a} p_t^a \qquad \text{(estimated power level)}$$

These tuples are then sorted by the mean power consumption. Let G_0^a denote the initial ordered set of tuples. In every iteration, two tuples are merged into one, effectively reducing the number of states by one. Those tuples are merged whose merging creates the smallest increase in quantization error. It is obvious to see that only merging two consecutive tuples can create a minimal increase in quantization error. The estimated new power level and the increase in quantization error are defined as:

$$p_{new}(g_i^a, g_j^a) = \frac{\tau_i^a \rho_i^a + \tau_j^a \rho_j^a}{\tau_i^a + \tau_j^a}$$
$$merge\_err(g_i^a, g_j^a) = |(\rho_i^a - p_{new}(g_i^a, g_j^a))|\tau_i^a + |(\rho_j^a - p_{new}(g_i^a, g_j^a))|\tau_j^a$$

In order to obtain G_{k+1}^a from G_k^a, $i, j = \underset{i,j}{\operatorname{argmin}}\ merge\_err(g_i^a, g_j^a)$ with $g_i^a, g_j^a \in G_k^a$ is sought but as already stated above, since G_k^a is a sorted set (by power level) only merging consecutive tuples can produce a minimal error, thus $i = \underset{i}{\operatorname{argmin}}\ merge\_err(g_i^a, g_{i+1}^a)$ identifies the tuples whose merging leads to a minimal increase in quantization error. Tuples are merged and so ultimately states are joined but the quantization error may not exceed a certain threshold.
The merge operator for two tuples is defined as

$$\begin{aligned} merge(g_i^a, g_j^a) = g_{new}^a = (\rho_{new}^a, \tau_{new}^a) \qquad & \text{with} \\ \rho_{new}^a = p_{new}(g_i^a, g_j^a) & \\ \tau_{new}^a = \tau_i^a + \tau_j^a & \end{aligned}$$

Thus ultimately,

$$\begin{aligned} G_{k+1}^a = (G_k^a \backslash \{g_i^a, g_{i+1}^a\}) \cup \{g_{new}^a\} \qquad & \text{with} \\ i = \underset{i}{\operatorname{argmin}}\ merge\_err(g_i^a, g_{i+1}^a) & \\ g_{new}^a = merge(g_i^a, g_{i+1}^a) \qquad & \text{and} \\ g_i^a, g_{i+1}^a \in G_k^a & \end{aligned}$$

After merging g_i^a with g_{i+1}^a, they are both removed from the sorted set and g_{new}^a is put into their position. No resorting is required since it holds that $\rho_i^a \leq \rho_{new}^a \leq \rho_{i+1}^a$.

A time series can be reconstructed using the quantized power levels from the set of tuples G_k^a. Let $r(G_k^a)$ be the reconstruction with power levels in

G_k^a, then

$$r(G_k^a)_{e_t:e_{t+1}} = \rho_i^a \qquad \text{with}$$

$$i = \underset{i}{\operatorname{argmin}} \; |\rho_i^a - [(e_{t+1}^a - e_t^a)^{-1} \sum_{l=e_t^a}^{e_{t+1}^a} p_l^a]|$$

The quantization error is defined as

$$q\_err(G_k^a, p^a) = \frac{\sum_t |r(G_k^a)_t - p_t^a|}{\sum_t p_t^a}$$

For real world appliances, empirical experiments have shown that as a stopping criterion $\epsilon = q\_err(r(G_0^a), p^a) + 0.035$ seems to yield good results.

In a last step every global event is associated with a state transition for each appliance. Note that state transitions are appliance specific. The pre- and post-event state are determined by calculating:

$$\text{pre-event state:} \qquad s_{n-1}^a = \underset{i}{\operatorname{argmin}} \; |\rho_i - [(e_n^g - e_{n-1}^g)^{-1} \sum_{t=e_{n-1}^g}^{e_n^g} p_t^a]|$$

$$\text{post-event state:} \qquad s_n^a = \underset{i}{\operatorname{argmin}} \; |\rho_i - [(e_{n+1}^g - e_n^g)^{-1} \sum_{t=e_n^g}^{e_{n+1}^g} p_t^a]|$$

Algorithm 1 shows how the algorithmic ground truth creation works in pseudo-code.

Data: single appliance measurement p_t^a, local event detector $local\_ed$, global events E^g
Result: state power levels
$E^a = (E^g \cap local\_ed(p_t^a)) \cup \{0, T\}$;
$G_0^a = \{g_0^a, ..., g_{N-1}^a\}$ following (1);
sort G_0^a by ρ_n^a;
$k = 0$;
while $q\_err(G_k^a) < \epsilon$ **do**
    $i = argmin\ merge\_err(g_i^a, g_{i+1}^a)$;
    $g_{new}^a = merge(g_i^a, g_{i+1}^a)$;
    $G_{k+1}^a = G_k^a \backslash \{g_i, g_{i+1}\}$;
    add g_{new}^a to G_{k+1}^a at former position of g_i;
    $k = k + 1$;
end
return G_{k-1}^a ;

Algorithm 1: Algorithmic ground truth creation.

4.3 Results

The local and global event detectors employed compute the second difference of the respective signals. The standard deviation σ of the second difference is computed and if the second difference exceeds the threshold of 2σ and 0.2σ for the local and global detector respectively, the time stamp is marked as an event.
In order to evaluate the performance of the ground truth creation system, sixteen appliances were quantized and the quantization errors, mean deviation (MD) between the quantized time series and actual measurements, as well as the detected number of states and the pre_error are reported. The pre_error shows the error that would have resulted when not merging any states but still approximating every intra-event power level by the mean. Table 4.1 shows the results.

The objective of the ground truth creation system is to keep the quantization error as low as possible while at the same time also pushing the number of inferred states low. The algorithm presented here seems to perform very well considering that on average the quantization introduced an error of 3.5% while approximating all appliances with 2-4 power levels.

Appliance	MD	q_err	#states	pre_error
Desk Lamp LR	0.03	0.01	2	0.011
DVR AV Blueray	0.23	0.01	2	0.004
Empty LR Socket	0.08	0.08	2	0.078
Hair Dryer	0.09	0.04	2	0.028
Iron	0.09	0.05	3	0.045
Kitchen Aid Chopper	0.09	0.06	3	0.044
LCD Monitor	0.17	0.03	2	0.0227
Monitor 2	0.18	0.04	4	0.0293
Printer	0.23	0.03	2	0.0156
Sub Woofer LR	0.18	0.02	2	0.011
Air Compressor	0.08	0.07	2	0.058
A-V LR	0.24	0.01	2	0.006
Garage Door	0.13	0.02	4	0.0167
Refrigerator	1.29	0.03	3	0.0263
Tall Desk Lamp LR	0.04	0.01	2	0.0076
TV Basement	1.0	0.03	2	0.0217

Table 4.1: The results of the quantization algorithm that extract power levels and event from single appliance measurements.

However, the algorithm struggles with appliances for which the assumption does not hold that the power trace is a piece-wise constant step function. The power consumption of some appliances gradually decrease over time which poses a problem to the algorithm. Figure 4.1 shows a plot of the actual power consumed by a fridge (top) and a garage door (bottom). The power consumption of the fridge during a freezing cycle decreases gradually and the algorithm makes small errors since it assumes constant power during a state. In other cases like e.g. the garage door, the power consumption is constant over a single activation cycle but not constant over different cycles. This might be due to some external factor like e.g. weather conditions. In this case the quantization will also struggle since it will assume constant power consumption over time.

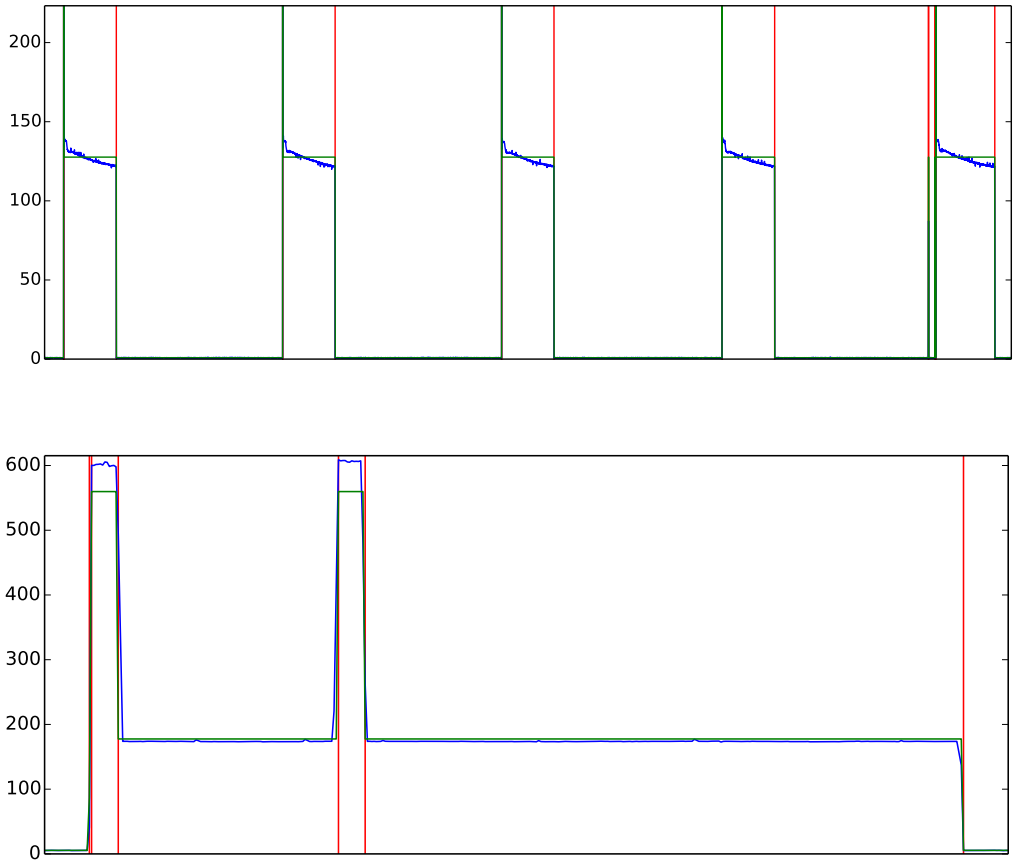

Figure 4.1: Quantization results for an activation cycle of a fridge (top) and a garage door (bottom). Vertical red lines denote state changes, the green graph denotes the quantization results and the blue graph denote the actual power consumption.

Chapter 5

Disaggregation by state inference

As already stated above, the model introduced here tries to bridge the gap between event-less and event-based approaches. It is a type of Factorial Hidden Markov Model but the energy level is not explained at every point in time but only at events. It is assumed that the aggregate power remains constant between events and that at every event maximal one appliance can change its state. Furthermore it is assumed that appliances emit high-level features $f(e_n)$ at every event e_n and that they jointly emit a sequence of power readings that we observe as the aggregate signal $\{p_{e_n}, ..., p_{e_{n+1}}\}$. The length of the sequence is variable and depends on when the next event is detected. Thus every HMM has two emissions: high-level features $f(e_n)$ and a contribution to the aggregate power trace $\{p_{e_n}, ..., p_{e_{n+1}}\}$. See Figure 5.1 for a graphical representation of the model.

5.1 Mathematical definitions

In our dual-emission FHMM, two separate sequences of different lengths and dimensionality, namely $p \in \mathbb{R}^{T \times q}$ and $f \in \mathbb{R}^{N \times r}$, are observed. Here, T and N are the lengths of the aggregate power sequence p and the changepoint feature vector f, respectively, whereas q and r are the dimensionality of each element in the respective sequences. The changepoint feature matrix can be filled by appending the outputs of a feature extractor applied to all detected events. These two sequences are conditionally independent given the hidden

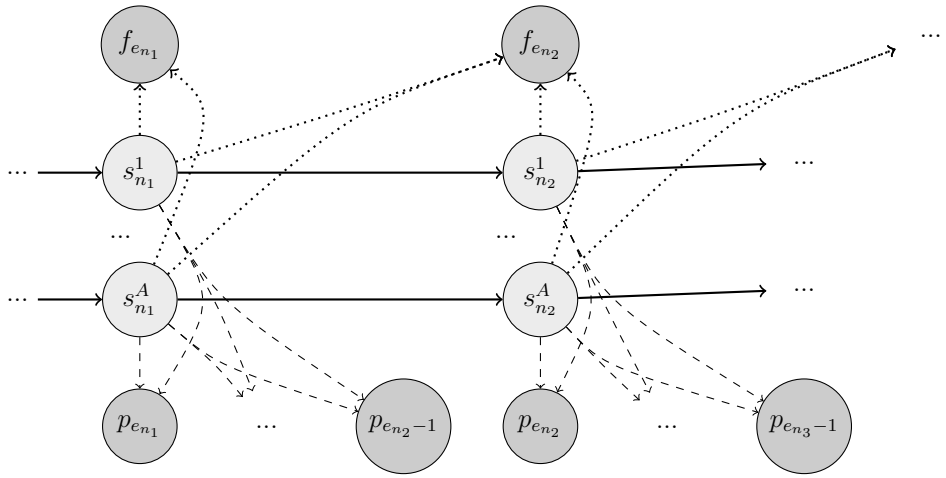

Figure 5.1: A graphical representation of the model. The number of points in time between change points is variable.

states of all of the HMMs in the factorial model. This is better illustrated in Figure 5.1. The motivation behind the two emissions is that in addition to the main observations of power p, there can be other (conditionally) independent observations occurring at changepoints such as environmental sensor measurements (e.g., sound, light intensity, etc.) or additional power measurements at a different resolution than p.

Unlike general FHMMs, the individual HMMs in our model are only allowed to transition one at a time, and only at changepoints of the p sequence. Thus, the model only makes sense once the observation sequence has been segmented by a changepoint detector. Let $E = \{0, e_1, ..., e_N, T\} \subset \{0, ..., T\}$ denote the ordered set of changepoints (0 and T are added simply for algorithmic convenience). The hidden states of each HMM independently emit features f at every changepoint depending on the previous and current state, and they jointly emit a sequence of power observations p up until the next changepoint depending solely on the current state. It is important to note that because the segmentation occurs beforehand, our model is no longer generative but rather discriminative (i.e., we do not model the length of the individual power segments emitted at each state). Traditional HMMs only allow geometric state durations. This model however is agnostic about state durations. In [11] a generalization of HMMs is introduced that specifically models the state durations and constrains the durations in such a way that

they stem from a particular distribution. In this work however, the state durations are not constrained in any way for two reasons: 1) The ultimate goal of a NILM system is to save energy by providing feedback to the user. Energy can only be saved if there is a saving potential and a saving potential can either be that an appliance is turned on longer than it should be or that it consumes more power than it should. Since the mean power consumption of an appliance is assumed to be constant, the model cannot identify when an appliance uses more power than it should. The only saving potential that can be exploited is when an appliance is turned on longer than it should. If however, the state durations are modeled explicitly then identifying those saving potentials might also prove harder. 2) Modeling the state durations explicitly increases the computational cost of inference significantly since looping over different durations is then required.

Let $s \in \mathbb{N}^{N \times A}$ be a state matrix with $N = |E|$ being the number of changepoints and A the number of HMM chains. s_n^a denotes the state of HMM a after event e_n. Then the conditional probability $P(s|f,p)$ has the following proportionality:

$$P(s|f,p) \propto P(s_0) \prod_{n=1}^{N-1} (\prod_{a=0}^{A} \underbrace{(P(s_n^a|s_{n-1}^a))}_{\text{state transition}} \underbrace{P(f(e_n)|s_n^a, s_{n-1}^a))}_{\text{event emission}} \underbrace{P(p_{e_n}, ..., p_{e_{n+1}-1}|s_n)}_{\text{joint emission}} \tag{5.1}$$

$P(s_0)$ is probability distribution over initial joint states. $P(s_n^a|s_{n-1}^a)$ denotes the probability of HMM a transitioning from state s_{n-1}^a into state s_n^a. $P(f(e_n)|s_n^a, s_{n-1}^a)$ models the probability of the features $f(e_n)$ observed at the change point detected a time point e_n given a state transition from s_{n-1}^a to state s_n^a. $P(f(e_n)|s_n^a, s_{n-1}^a)$ allows the model to examine quantities of the state change and make a guess about its nature [1]. Inferring a state transition rather than just the current state is motivated by the application domain: if e.g. a hair dryer transitions into the *high* position from the *medium* position, commonly used features such as the increase in power consumption will be very different from turning the hair dryer from *off* to *high*. $P(f(e_n)|s_n^a, s_{n-1}^a)$

[1]Note that during a state transition, the model assumes all HMMs emit the same feature. In the setting of energy disaggregation this is of course non-sensical: One appliance state change is responsible for features and other appliances do not emit these features by remaining in the same operational state. But modeling feature emission like this allows for plugging in probabilistic classifiers from the event-based energy disaggregation community.

can be interpreted as the Bayesian inversion of the output of a probabilistic classifier trying to infer the state transition given the observed features. Thus, e.g. the likelihood term of a Naïve Bayes classifier could in principle be used as a model. $P(p_{e_n}, ..., p_{e_{n+1}}|s_n)$ holds the joint state s_n jointly accountable for the observed joint features from the current possible change point up until the next. Let ρ_i^a be the estimated power consumption of appliance a in state i. Sensible models will penalize the deviation of $\sum_a^A \rho_{s_n^a}^a$ from the observed power features $p_{e_n}, ..., p_{e_{n+1}}$. There are multiple possible ways to model this distribution.

5.1.1 Event-based and event-less aspects

The *event emission* part of the model can be considered an event-based approach. The model iterates over the events provided by the event detector. For the model of $P(f(e_n)|s_n^a, s_{n-1}^a)$ any probabilistic classifier can be plugged in. It takes the features of the feature extractor f as input and outputs the probability that these features are emitted when appliance a changes states from s_{n-1}^a to s_n^a. The *joint emission* part of the model can be used to force the model to trace the observed signal. If it is e.g. modeled by

$$P(p_{e_n}, ..., p_{e_{n+1}}|s_n) = \prod_{t=e_n}^{e_{n+1}} N(\sum_{a=0}^{A} \rho_{s_n^a}^a + \Sigma^{1/2} z_t, \Sigma)$$

the additive FHMM used in [13] would be mimicked.
So ultimately, this general model ties event-based and event-less approaches together and offers a framework to plug in event detectors, feature extractors, classifiers from event-based approaches and allows for combining them with energy estimators from event-less approaches.
This lets us formulate a disaggregation problem as a 6-tuple $D = (E, \pi, S, F, C, J)$ with

- Event detector: $E : ed(p) \rightarrow \{e_0, ..., e_N\}$ a function that takes the aggregate signal as input and outputs time points of significant changes in power. These time points should ideally allow to model the power traces of individual appliances with minimal error.

- Initial state prior: $\pi : P(s_0)$ probability distribution over an assignment of an initial state to every appliance

- State transition probabilities: $S : P(s_n^a = i | s_{n-1}^a = j)$ for every appliance a.
- Feature extractor: $F : f(e_n) \rightarrow \mathbb{R}^m$ a function that takes an event as input and outputs any quantity that might shed light on the identity of the state transition. These features can include high-level features such as harmonics or external features, such as time, outside temperature, ...
- Probabilistic classifier: $C : P(f(e_n) | s_n^a, s_{n-1}^a)$ a classifier that outputs the probability of the features being emitted by a state transition.
- Energy estimator: $J : P(p_{e_n}, ..., p_{e_{n+1}} | s_n)$ a model that explains the observed power trace up until the next event given the current transition

5.2 Efficient approximate inference

Exact inference in Factorial Hidden Markov Models requires to iterate over exponentially many states. The basic Viterbi algorithm is a backwards-looking algorithm that for every point in time iterates over all possible states and infers the most likely path ending in this state. It is backwards-looking in the sense that when computing the paths ending at time t, it iterates over all most likely paths ending at time $t-1$ and computes the probability of transitioning from the previous to the next state. Mathematically speaking: Let $\phi(i, t-1)$ denote the probability of the most likely path ending in state i at time point $t-1$, then for $\phi(j, t)$ it holds that $\phi(j, t) = \max_i \phi(i, t-1) P(j|i)$. Thus for every point in time computations in $\mathcal{O}(k^2)$ are required. This is only tractable for a small number of possible states.
Exact inference in Factorial Hidden Markov Models would require to consider all combinations of possible states for all hidden chains which are for A many appliances with k many states k^A, thus for every point in time computations in at least $\mathcal{O}(k^{A+1})$ would need to be carried out which becomes intractable quickly. This problem is tackled by first, making a simplifying assumption and by second, cutting off the search space.
The simplifying assumption that is made is a variant of the *one-at-a-time* assumption that is also made in [13]. The original assumption states that at any point in time, only a single appliance can change its state. In this case, it is assumed that at any event, only a single appliance can change its state.

Since events are time stamps and possibly all time points can be events, the assumption made here is not necessarily stronger.
The Viterbi algorithm can be reformulated as a forward-looking algorithm that updates the probabilities of all successor states given a previous state. Let

$$suc(i) = \{1, ..., k\}$$
$$P(j|i) = P(s_n = j|s_{n-1} = i)P(p_{e_n}, ..., p_{e_{n+1}}|s_n = j)P(f(e_n)|s_n = j, s_{n-1} = i)$$

with

$$P(f(e_n)|s_n = j, s_{n-1} = i) = \prod_a^A P(f(e_n)|s_n^a = j^a, s_{n-1}^a = i^a)$$

Input: $\phi(i, t-1)$
Output: $\phi(i, t)$
`// initializing state probabilities`
for $j \in \{1, ..., k\}$ **do**
  $\phi(j, t) = -\infty$;
end
`// Computing successor probabilities`
for $i \in \{1, ..., k\}$ **do**
  for $j \in suc(i)$ **do**
    $\phi(j, t) = max(\phi(j, t), \phi(i, t-1)P(j|i))$;
  end
end

Algorithm 2: Forward-looking Viterbi

Algorithm 2 is equivalent to the standard Viterbi algorithm if the set of all successor states is the set of all states itself but it allows for reducing the computational cost dramatically if the number of successor states can be pruned.
The *one-at-a-time* constraint allows exactly for this: Assume that appliance a has k_a-many states and also assume that $s_n \in \mathbb{N}^A$ is a vector containing an assignment of a state for every appliance. The number of successor states of s_n **without** the constraint is $\prod_a^A k_a$ but **with** the constraint the number can be reduced to $(1 - a) + \sum_a^A k_a$. The successors of s_n can be computed by Algorithm 3

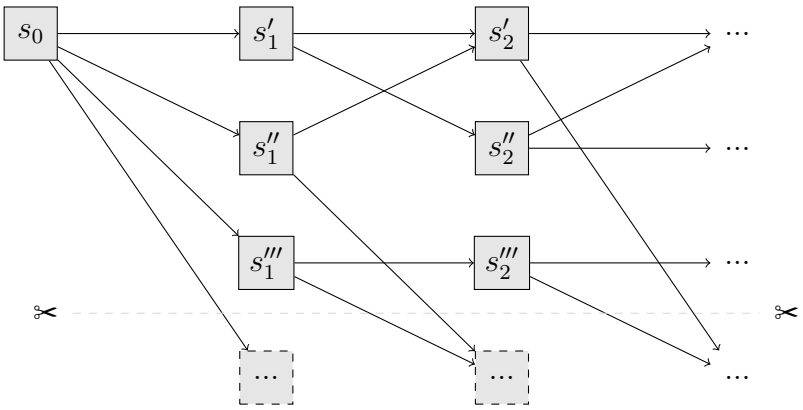

Figure 5.2: Search space reduction by cut-off. All nodes are ordered by their probability, thus $\phi(s'_1, 1) > \phi(s''_1, 1) > \phi(s'''_1, 1)$ and $\phi(s'_2, 2) > \phi(s''_2, 2) > \phi(s'''_2, 2)$

Input: s_n
Output: set of successors S
$S = \{s_n\}$;
`// Computing successors`
for $a \in \{1, ..., A\}$ **do**
  for $j \in \{1, ..., k_a\}$ **do**
    `// Avoiding multiple` s_n`'s in output`
    if $s_n^a \neq j$ **then**
      $s_{n+1} = s_n$;
      $s_{n+1}^a = j$;
      $S = S \cup \{s_{n+1}\}$;
    end
  end
end

Algorithm 3: Computing all successor of a given state

The *one-at-a-time* constraint allows to reduce the number of successors state of a given state. If however, at every event all successors of all states are computed, the number of possible states will still grow exponentially. That's why the search space needs further restrictions: at every event only the b most probable paths so far are considered further. Figure 6.1 shows how states are cut off. In this example $b = 3$ and the paths are sorted by their respective probability, thus $\phi(s'_1, 1) > \phi(s''_1, 1) > \phi(s'''_1, 1)$

Both restrictions allow for efficient approximate inference in FHMM which is described in Algorithm 4. From a search perspective, the algorithm is a type of beam search, i.e. a breadth-first search with a limited agenda.

Input: s_0
Output: most probable sequence $s_1, ..., s_N$
$agenda = \{s_0\}$;
initialize $\phi(s,t) = -\infty$ for all s and t;
$\phi(s_0, 0) = 1$;
for $n \in \{1, ..., N\}$ **do**
  for $i \in agenda$ **do**
    `// one-at-a-time assumption incorporated into` $suc(i)$
    for $j \in suc(i)$ **do**
      $\phi(j,n) = max(\phi(j,n), \phi(i,n-1)P(j|i))$;
    end
  end
  `// cut-off restriction`
  $agenda = b$-best states in $\phi(*, n)$
end
e

Algorithm 4: Efficient inference in FHMM

Chapter 6

Results

This thesis on one hand introduces a novel inference technique for factorial Hidden Markov Models and on the other hand an augmentation in order to incorporate transient information. Both novelties will be evaluated individually. In a first experiment, a synthetic dataset is created and the performance of the inference technique will be compared to AFAMAP (based on Integer Programming) and Structured Mean Field (based on variational methods). The second experiment ought to show that incorporating transient information increases the disaggregation performance of the system. The second experiment is conducted on the data set introduced in [2].

6.1 Experiment 1

The goal of the first experiment is to show that the inference technique introduced in this work can enable the trade-off between computational time and disaggregation accuracy and that ultimately, the modified Viterbi algorithm outperforms Structured Mean Field and AFAMAP. AFAMAP is an inference introduced in [13] based on Integer Programming. For this, a synthetic dataset is created. The synthetic data set that is used is the same as in [13]. The dataset contains 20 sets that each comprise 10 cyclic HMMs, each with 4 states and 4 dimensional output. The initial state of each HMM is drawn from a uniform distribution. The mean output of each HMM in every state is drawn from a uniform distribution in the range $[0, 2]$. The observation is the sum of all HMMs plus zero-mean Gaussian noise with a covariance of $0.01I$.

6.1.1 Experimental setup

For the first experiment, no transient information is incorporated. Thus the model of $P(s_n, s_{n-1}|f(e_n))) = 1$ for all $f(e_n)$, s_n and s_{n-1}. The model $P(p_{e_n}, ..., p_{e_{n+1}}|s_n)$ simply penalizes any deviation of the observed from the inferred aggregate power:

$$P(p_{e_n}, ..., p_{e_{n+1}}|s_n) = \prod_{e_n}^{e_{n+1}} L(p_{e_n} - \sum_{a}^{A} \rho_{s_n}^{a}|0, 0.5)$$

with $L(x|\mu, b) = \frac{1}{2b} \exp\left(-\frac{|x-\mu|}{b}\right)$ being a Laplace distribution. Since *AFAMAP* incorporates some transient information in the difference model, the fairest comparison is to *AFAMAP no diff* where the difference model is deactivated. The model parameters for the model introduced in this work (which we will refer to as DBSI) were obtained using the algorithmic ground truth creation introduced earlier on a held out training set. The model parameters for *SMF* and *AFAMAP no diff* are the **true distribution in the data** which should give these approaches an advantage.

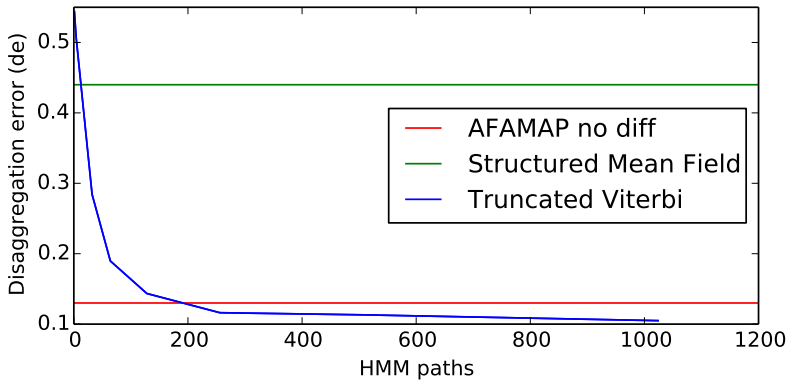

Figure 6.1: Even in a scenario where *SMF* and *AFAMAP no diff* are provided with a higher amount of ground truth, the inference technique introduced here outperforms its competitors

The parameter that controls how many HMM paths are cut off is varied in the experiment. The results are depicted in Figure 6.1. Even though the competitors have an advantage due to having access to the true underlying distributions during inference, DBSI still outperforms them given that

enough HMM paths are kept. The inference technique introduced here allows to control the computational expense and can, given unlimited computational power, approximate exact inference. The computational time increases however linearly with the number of retained HMM paths.
The truncated Viterbi algorithm introduced here also has advantages from a modeling point-of-view: DBSI is in a sense a meta-model. Different models for the feature emission and joint emission can be plugged in in order to adjust the model to different scenarios. If for example the model has to cope with unmodeled devices, the joint emission can be extended with a component that soaks up residual power that cannot be explained by the devices that are modeled. Variational methods as well as Integer Programming techniques for inference require reformulating the inference problem, i.e. for variational methods the analytical solution to minimizing the Kullback-Leibner divergence needs to be calculated whereas the inference technique based on the truncated Viterbi algorithm allows for changing the underlying model components without changing the inference technique.

6.2 Experiment 2

The goal of the second experiment is to show that incorporating transient information can improve the disaggregation performance. For this, single appliance measurements of nine appliances over the span of 5 days were summed up and then subsequently disaggregated. The data collection system is described in [2]. Figure 6.2 shows the power levels and the energy that these appliances consumed. One should note that the aggregate signal was not measured but instead the artificial sum was disaggregated. This is due to the problems discussed in section 3. One should also note that DBSI cannot exploit its full potential in this scenario: the appliances were measured at a sampling rate of roughly 1.5Hz, i.e. the feature extractor cannot make use of harmonics (high frequency information). Furthermore no external features like e.g. outside temperature, time of the day or side-channel information like e.g. light intensity were made use of.

Appliance	Power Levels	Energy
Compressor	1100W	104.9Wh
A-V	41.5W	2321.7Wh
Desk lamp	11.6W	220.3Wh
DVR/Blueray	33.5W, 75.3W	3861.2Wh
Garage door	86.6W, 177.5W, 559.7W	659.4Wh
Iron	923.6W, 1450.3W	142.0Wh
Fridge	127.8W, 420W	3546.8Wh
Tall desk lamp	23.7W	450.6Wh
TV	198W	3697.4Wh
total		15004.6

Figure 6.2: The appliance used in the disaggregation experiments with the algorithmically inferred power levels and overall energy consumption.

6.3 Instantiating the model

As already stated above, the model introduced in this work can be understood as a meta-model that allows for plugging in event detectors, state priors, state transition probabilities, feature extractors, probabilistic classifiers and energy estimators. This section shows an exemplary instantiation of this meta-model for the disaggregation of real world data.

6.3.1 Event detector

The same event detector that was used for ground truth creation is employed as an event detector. The event detector computes the second difference of the signal and if the second difference exceeds a threshold, an event is detected. Algorithm 5 describes the event detector in pseudo-code.

Input: power trace p, threshold ϵ
Output: set of events $E = \{e_1, ..., e_N\}$
$E = \{\}$;
$d_t = (p_{t+1} - p_t) - (p_t - p_{t-1})$;
$\sigma = std(d_t)$;
for $t \in \{1, ..., T-1\}$ **do**
  if $d_t < \epsilon\sigma$ **then**
    $E = E \cup \{t\}$
  end
end

Algorithm 5: A very simple event detector.

6.3.2 Initial state prior

Algorithm 4 for efficient inference in FHMMs requires as input an assignment of a starting state for every appliance: s_0. In traditional HMMs mostly a uniform prior over the starting states is assumed. This is for FHMMs however computationally expensive since then the computation of $\phi(*, 1)$ would require iteration over exponentially many states. The computational time could however be sacrificed because it would only need to be done once. The problem of inferring s_0 could also be tackled by simply waiting until most appliances are turned off and then assuming $P(s_0 = [0]^A) = 1$. In this work it is assumed that s_0 is known, i.e. the state s_0 is taken from ground truth. However experiments have shown that initializing the model simply with $P(s_0 = [0]^A) = 1$ without actually waiting until the aggregate signal reaches a minimum has very little effect on the performance. Another strategy that seems feasible but was not tested is to infer a distribution over joint starting states. One would scan the training set for all observed state combinations. The set of all observed state combinations is e.g. on the data set used for evaluation 67 opposed to 2304 possible joint states.

6.3.3 State transition probabilities

The algorithmic ground truth creation described in section 4 was used in conjunction with the event detector described above to infer event labels for the training set. These event labels can then be used to estimate the set of state transition probabilities for every appliance. Let c be a counting

function, then

$$P(s_t^a = j | s_{t-1}^a = i) = \frac{c(s_{t-1}^a = i, s_t^a = j)}{c(s_t^a = i)}$$

6.3.4 Feature extraction

The feature extractor used in this work computes the first difference of the signal in a window of 10 samples around the event and the resulting vector is then projected onto a lower dimensional space using Principal Component Analysis retaining 93% of the variance. The projection matrix is learned on the held-out training set. In this case, the feature extractor was mainly limited by the nature of the available data. The data provided did not include high frequency current and voltage readings which would be required for the computation of active and reactive power or harmonics. Merely apparent power was provided in the dataset. Also since the data was recorded over only consecutive five days, features like the season or outside temperature become pointless.

6.3.5 Probabilistic classifier

As already stated above any probabilistic classifier can be plugged into $P(f(e_n)|s_{n-1}, s_n)$. Since $f(e_n)$ is a multivariate and continuous variable $P(f(e_n)|s_{n-1}, s_n)$ cannot be estimated by simply counting occurrences. A model that generalizes over $f(e_n)$ is a necessity. As an example, a Gaussian Naïve Bayes will be used to model $P(f(e_n)|s_{n-1}, s_n)$ but in theory any type of probabilistic classifier can be employed.
It is assumed that every feature value $f(e_n)_i$ (remember that $f(*)$ maps into $\mathbb{R}^m$, so $f(e_n)_i$ denotes the ith element of $f(e_n)$) stems from a conditionally independent Gaussian distribution:

$$P(f(e_n)|s_{n-1}^a, s_n^a) = \prod_i P(f(e_n)_i|s_{n-1}^a, s_n^a) \qquad \text{with}$$
$$P(f(e_n)_i|s_{n-1}^a, s_n^a) = N(f(e_n)_i|\mu_{s_{n-1}^a, s_n^a}, \sigma_{s_{n-1}^a, s_n^a})$$

A maximum likelihood estimation of $\mu_{s_{n-1}^a, s_n^a}$ and $\sigma_{s_{n-1}^a, s_n^a}$ is straight-forward: $\mu_{s_{n-1}^a, s_n^a}$ and $\sigma_{s_{n-1}^a, s_n^a}$ can be obtained by computing the variance and mean of those instances of $f(e_n)$ which mark a transition from s_{n-1}^a into s_n^a.

6.3.6 Energy estimation

After the ground truth creation of section 4 was applied to the training data, an estimate of the power levels for every appliance in every state is provided. Let the estimate of the power consumption of appliance a in state i be ρ_i^a. The ground truth creation also provides ground truth on events and their labels. Since the ground truth creation assumes piece-wise constant energy consumption of appliances, the sum of estimates deviates from the observed aggregate signal. Let the time series of residuals $r_t = \hat{p}_t - p_t$

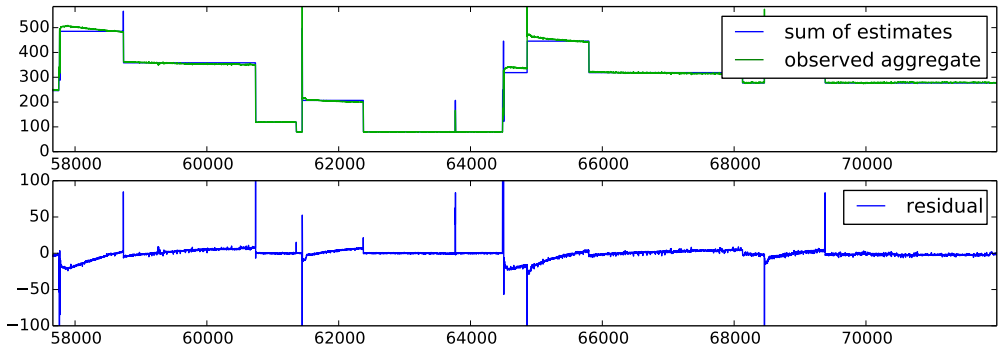

Figure 6.3: The residuals or quantization error introduced by the algorithmic ground truth creation

One would assume that ideally $r_t \approx 0$ but as Figure 6.3 shows, such a residual time series contains much structure and simply minimizing r_t would lead the model to make mistakes in that it would for example squeeze in a low power consuming appliance into the transient of a high power appliance. This is why an AR (autoregressive) model of the type $r_t = \alpha_0 + \alpha_1 r_{t-1} + ... + \alpha_n r_{t-n}$ is trained to predict the residual out of its autostructure. Then, the deviation from the prediction is penalized. This can be understood as alleviating the i.i.d. assumption on the residual since the dependence of the current error on previous errors is incorporated. Let $\chi(r_t) = \alpha_0 + \alpha_1 r_{t-1} + ... + \alpha_n r_{t-n}$,

then

$$P(s_n|p_{e_n},...,p_{e_{n+1}}) = \prod_{t=e_n}^{e_{n+1}} L(r_t|\chi(r_t),\sigma) \qquad \text{with}$$

$$L(x \mid \mu, b) = \frac{1}{2b}\exp\left(-\frac{|x-\mu|}{b}\right) \qquad \text{and}$$

$$r_t = p_t - \sum_a^A \rho^a_{s^a_n}$$

6.4 Outcome

Table 6.4 compares the performance of DBSI with and without the second emission. The dataset was separated into 5 chunks of 100.000 data points ($\approx$ 18.5 hours) each. Everything but one chunk was used as a training set and the power consumption of the appliances was disaggregated on that chunk. Then the predicted power consumptions were appended in such a way that the original whole aggregated power is reconstructed. One might think that evaluating the chunks individually makes more sense because then the variation of the performance could be reported. However, this leads to the problem that the energy of a number of appliances in more than one chunk is zero and since most performance criteria divide by the energy, this would lead to a division by zero.

The overall performance increases by 10% by incorporating the second emission. However, the performance for some appliance actually decreases by adding the feature emission. This can be explained by the nature of the data: when the second emission is incorporated, DBSI confuses the Compressor and the Iron. Both appliances have very similar power levels and the problem is that the activity in the Iron time series is very clustered. In the first 400.000 data points the Iron was not used at all, then in the next 100.000 data points the Iron was turned on and off 16 times and in the remainder of the time the Iron was turned on twice. Thus, when the algorithm evaluates the performance on the 5th chunk of data, the training set for the classifier consists of two events each for $on \rightarrow off$ and $off \rightarrow on$ transitions. During training the parameters of the Naïve Bayes likelihood term need to be estimated. For each feature dimension there are two parameters (μ and

	Actual	DBSI			no 2nd emit		
Appliance	Observed E	Estimated E	F1	DE	Estimated E	F1	DE
Compressor	104.9Wh	109.6Wh	0.73	0.73	104.77Wh	0.96	0.96
A-V	2321.7Wh	2240.2Wh	0.98	0.97	2301.4Wh	0.90	0.90
Desk lamp	220.3Wh	280.5Wh	0.87	0.88	219.4Wh	0.80	0.80
DVR/Blueray	3861.2W	3840.7Wh	0.96	0.96	3589Wh	0.94	0.95
Garage door	659.4Wh	591.5Wh	0.84	0.83	1899.8	0.50	0.04
Iron	142.0Wh	131.1	0.65	0.64	142Wh	0.97	0.97
Fridge	3546.8Wh	3535.8	0.96	0.96	3654Wh	0.87	0.86
Tall desk lamp	450.6Wh	575.8	0.87	0.88	511Wh	0.75	0.73
TV	3697.4Wh	3695.0	0.98	0.98	2583Wh	0.80	0.83
total	15004.6	15000.6		**0.95**	15006.7		**0.85**

Figure 6.4: The appliance level performance of the disaggregation system with and without a feature emission.

σ). But two parameters cannot be estimated properly from just two data points.
For appliances where there is a consistent and big number of events, the performance is boosted substantially. For all appliances that consumed more than 1kWh, the performance increased on average by roughly 5% measured by the disaggregation error. Figure 6.5 shows that the overall energy inferred for each appliance also improves by incorporating the second emission.
For this particular data set however, the algorithm could not unfold its full potential. External features or high frequency features could not be made use of. But on the other hand, this data set is very well behaved. There is no baseload (the power consumed by appliances that are always turned on like e.g. a smoke detector) and many data collection problems were circumvented by summing up the individual appliances rather than disaggregating a subdistribution measurement.

All experiments were performed by retaining the 50 most probable HMM paths. The question arises how many paths should be kept in a real world disaggregation scenario. Figure 6.6 shows the normalized log mean probabilities. These were obtained by sorting all paths at every event by their probability. In a next step, these probabilities were normalized for every event in such a way that the probability of the most probable path is 1 (or the log-probability is 0). Then the mean over all events was taken. It can be

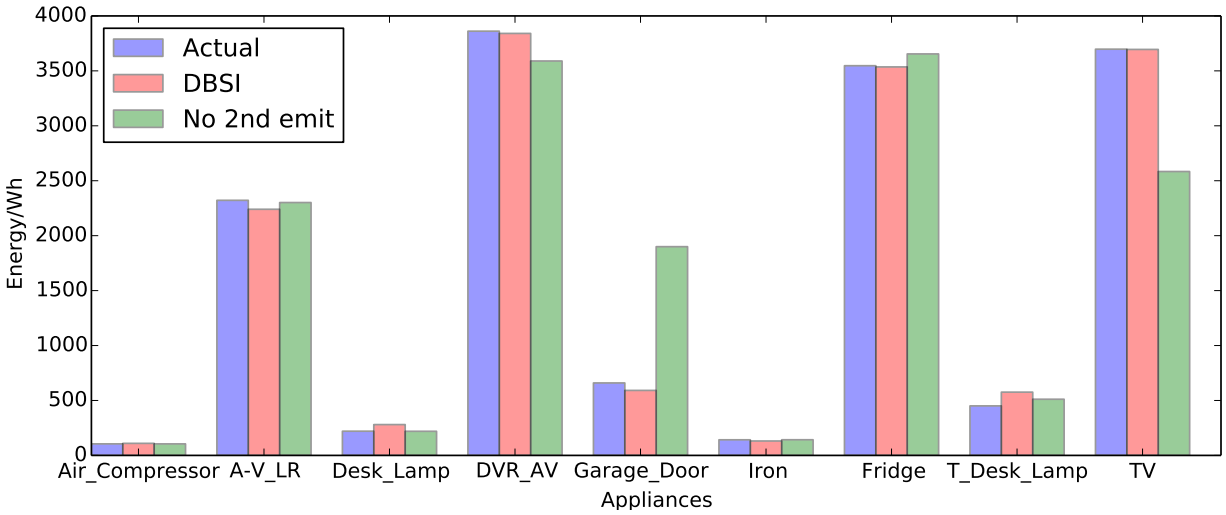

Figure 6.5: The energy inferred for each appliance. One can see that incorporating the feature emission into the model improves the performance.

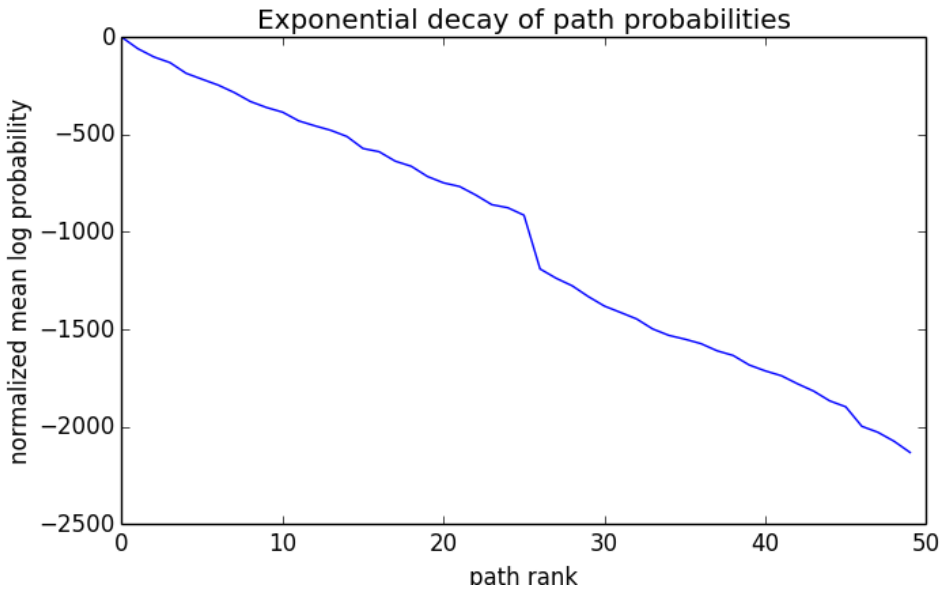

Figure 6.6: The normalized log-probability of the ranked most probable paths.

seen that the probabilities drop off exponentially (= the log probabilities fall of linearly) and that after the 25th retained paths, the steepness of the decay even increases. Analyzing the path of the most probable sequence revealed that at no event a path was chosen that was not amongst the 20 most probable options. This does of course not mean that if the parameter controlling the number of paths retained was increased, a more probable path could not be found that choses at some event an option that is ranked lower than 50.

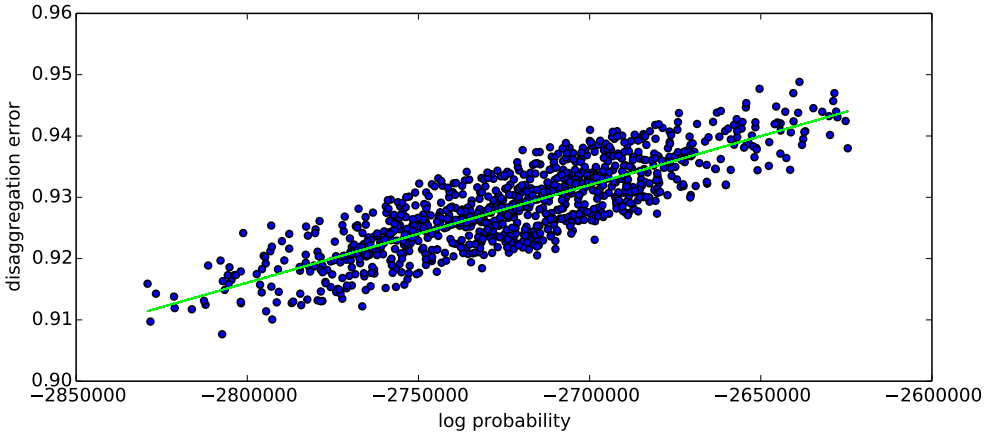

Figure 6.7: Log probability of a sequence plotted against the disaggregation error. The green line shows the regression.

A problem that sometimes arises especially in Natural Language Processing is that probabilities sometimes do not commensurate with the actual goal of the model. In the case of NILM, the probability of a state sequence is maximized but we are not interested in a high probability sequence but such a sequence that results in a maximal disaggregation performance. In order to see if a higher probability sequence also results in a sequence with higher disaggregation performance, the correlation between the two quantities was computed. The correlation between the two quantities is with 0.84 quite high. Figure 6.7 shows a plot of the log-probability of a sequence against disaggregation performance. It is easy to see that the two quantities seem to commensurate well.

Chapter 7

Conclusion

This work has made three contributions. First, a method that takes single appliance measurements as input and extracts power levels and events was introduced. Second, this work shows how event-based and event-less approaches can be fused to overcome limitations that both approaches have in isolation. Third, a generic inference technique is introduced that allows for efficient inference in that model.
The performance of the individual components was tested on a synthetic and a real world data set and both experiments yielded positive results. However, there is still much room for improvement. One should note that the experiment on the real world dataset circumvented some NILM problems by disaggregating the artificial sum rather than an actual measurement on the whole home or subdistribution level.
In order to commercialize NILM systems, the amount of ground truth required must be minimized. It is financially not viable to install a submetering system in a building to collect ground truth for a certain amount of time and then later remove that system again. This not only defeats the purpose of NILM systems but also assumes that no appliance is added to the building later. The system introduced here can however be made viable by three approaches:

1. The system could be employed in scenarios where appliances stay the same across multiple buildings. This is for example true for franchise chains.

2. Multi-task learning approaches could be employed that transfer knowledge of one building to another. This would still require huge amounts

of ground truth but after an initial very costly phase of collecting this ground truth, deploying the system becomes hopefully cheaper and cheaper.

3. The ground truth requirements could also be relaxed algorithmically. This system couples power levels with features. Event-based systems base their decision on event features whereas event-less systems base their decision on power levels mostly. The system introduced here enforces an agreement between features and power levels. This might add more rigidity to usually fragile unsupervised methods. An EM-like algorithm is conceivable since the E-step has become tractable thanks to the novel inference technique.

On top of that, there is also room for improvement on the side of the inference technique. One problem that arises is that state sequences with minimal differences might occupy the most probable paths. There are sometimes surges in the power line that only last for less than one second. The energy these spikes consume is negligible. A problem with simply retaining the most probable paths is that the most probable paths might actually be all the same and their only difference is their explanation of this single spike. A system that identifies maximally similar state sequences and removes or merges those paths might boost the performance of the system substantially.

However, the system introduced here could not unfold its full potential: there is currently no data set that contains side-channel information such as readings from a light or sound intensity meter. These additional information can easily be incorporated into the system. The feature extractor could simply take these additional quantities into account.

Bibliography

[1] U.S. Energy Information Administration. "Annual energy review 2009". In: *Technical report* (2009).

[2] Kyle D Anderson. "Non-Intrusive Load Monitoring: Disaggregation of Energy by Unsupervised Power Consumption Clustering". In: (2014).

[3] Leonard E Baum et al. "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains". In: *The annals of mathematical statistics* (1970), pp. 164–171.

[4] Christian Beckel et al. "The eco data set and the performance of non-intrusive load monitoring algorithms". In: *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*. ACM. 2014, pp. 80–89.

[5] Mario E Berges et al. "Enhancing electricity audits in residential buildings with nonintrusive load monitoring". In: *Journal of industrial ecology* 14.5 (2010), pp. 844–858.

[6] Sarah Darby et al. "The effectiveness of feedback on energy consumption". In: *A Review for DEFRA of the Literature on Metering, Billing and direct Displays* 486 (2006), p. 2006.

[7] Jesse Davis and Mark Goadrich. "The relationship between Precision-Recall and ROC curves". In: *Proceedings of the 23rd international conference on Machine learning*. ACM. 2006, pp. 233–240.

[8] Stuart Geman and Donald Geman. "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 6 (1984), pp. 721–741.

[9] Zoubin Ghahramani and Michael I Jordan. "Factorial hidden Markov models". In: *Machine learning* 29.2-3 (1997), pp. 245–273.

[10] George William Hart. "Nonintrusive appliance load monitoring". In: *Proceedings of the IEEE* 80.12 (1992), pp. 1870–1891.

[11] Matthew J Johnson and Alan S Willsky. "Bayesian nonparametric hidden semi-markov models". In: *The Journal of Machine Learning Research* 14.1 (2013), pp. 673–701.

[12] J. Z. Kolter, Siddharth Batra, and Andrew Y. Ng. "Energy Disaggregation via Discriminative Sparse Coding". In: *Advances in Neural Information Processing Systems 23*. Ed. by J.D. Lafferty et al. Curran Associates, Inc., 2010, pp. 1153–1161. URL: `http://papers.nips.cc/paper/4054-energy-disaggregation-via-discriminative-sparse-coding.pdf`.

[13] J Zico Kolter and Tommi Jaakkola. "Approximate inference in additive factorial hmms with application to energy disaggregation". In: *International conference on artificial intelligence and statistics*. 2012, pp. 1472–1482.

[14] J Zico Kolter and Matthew J Johnson. "REDD: A public data set for energy disaggregation research". In: *Workshop on Data Mining Applications in Sustainability (SIGKDD), San Diego, CA*. Vol. 25. Citeseer. 2011, pp. 59–62.

[15] Andrew J Viterbi. "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm". In: *Information Theory, IEEE Transactions on* 13.2 (1967), pp. 260–269.

[16] Peter Willett. "Recent trends in hierarchic document clustering: a critical review". In: *Information Processing & Management* 24.5 (1988), pp. 577–597.

[17] Michael Zeifman and Kurt Roth. "Nonintrusive appliance load monitoring: Review and outlook". In: *IEEE Transactions on Consumer Electronics* (2011), pp. 76–84.

[18] Ahmed Zoha et al. "Non-intrusive load monitoring approaches for disaggregated energy sensing: A survey". In: *Sensors* 12.12 (2012), pp. 16838–16866.